COPY

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Patent No.: 6,959,316 B2
Issue Date: October 25, 2005
Application No.: 09/726,188
Applicant(s): Jari A. Purviainen
Filed: November 29, 2000
Title: Dynamically Configurable Processor

Attorney Docket No.: 872.0025.USU
Customer No.: 29,683

Commissioner For Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Request For Certificate of Correction
(Office Mistake)
35 U.S.C. §254 (37 C.F.R. §1.322)

Sir:

This is a request for a Certificate of Correction (MPEP 1480) in regard to the above-identified patent. Attached is a Form PTO-1050. The mistakes, incurred through the fault of the Patent and Trademark Office, are clearly disclosed by the records of the Office as indicated by the following description:

In Claim 1: Column 8, line 39, after "sources" please add the following language:
--, wherein said plurality of ALUs, wherein the second mode of operation, operate in parallel with one another one data received from said second data sources, and wherein said data processor forms a part of a wireless terminal--. See Claim 1 of Examiner's Amendment dated May 24, 2005.

In Claim 8: Column 9, line 8, after "sources" please add the following language:
--, wherein said plurality of ALUs, wherein the second mode of operation, operate in parallel with one another one data received from said second data sources, and wherein said data

processor forms a part of a wireless terminal--. See Claim 10 of Examiner's Amendment dated May 24, 2005.

In Claim 17:

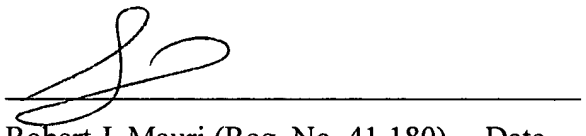
Column 10, line 22 please delete "ALU" and replace with –ALUs--.

Column 10, line 23, after "n-bits" please add the following language:

--, wherein said DSP forms a part of a wireless terminal --. See Claim 21 of Examiner's Amendment dated May 24, 2005.

The Office is requested to issue a Certificate of Correction.

Respectfully submitted,



11/7/05

Robert J. Mauri (Reg. No. 41,180) Date

Customer No.: 29683
Harrington & Smith, LLP
4 Research Drive
Shelton, CT 06484-6212
203-925-9400

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail on the date shown below in an envelope addressed to: Commissioner For Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

11/7/05
Date

Elaine F. Main
Name of Person Making Deposit

(Also Form PTO-1050)

UNITED STATES PATENT AND TRADEMARK OFFICE
CERTIFICATE OF CORRECTION

PATENT NO. : **6,959,316 B2**
DATED : **October 25, 2005**
INVENTOR(S) : **Jari A. Parviainen**

It is certified that an error appears or errors appear in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Claim 1: Column 8, line 39, after "sources" please add the following language:
--, wherein said plurality of ALUs, wherein the second mode of operation, operate in parallel with one another one data received from said second data sources, and wherein said data processor forms a part of a wireless terminal--.

Claim 8: Column 9, line 8, after "sources" please add the following language:
--, wherein said plurality of ALUs, wherein the second mode of operation, operate in parallel with one another one data received from said second data sources, and wherein said data processor forms a part of a wireless terminal--.

Claim 17:
Column 10, line 22 please delete "ALU" and replace with –ALUs--.
Column 10, line 23, after "n-bits" please add the following language:
--, wherein said DSP forms a part of a wireless terminal --.

MAILING ADDRESS OF SENDER (Please do not use customer number

Robert J. Mauri, Esq.
Harrington & Smith, LLP
4 Research Drive
Shelton, CT 06484-6212

PATENT NO. **6,959,316 B2**

No. of additional copies

P15/REV03

NOV 2 1 2005

In accordance with the teachings herein the final 32-bit addition unit **36** can be implemented as two, 16-bit ALUs, which are then available as well for other processing tasks besides multiplication. In addition, techniques exist for reducing the number of remaining addition units (see, for example, Peter Pirsch: "Architectures for Digital Signal Processing"). One possible approach is use the Carry/Save/Addition principle, where it is possible to calculate together three input operands and forward the carry signal to the next stage(s). This method enables one to optimize the number of addition units. Other techniques exist for implementing the multiplier, as well as for optimizing the required hardware.

In practice the multiplied values of A and B can have the same sign or different signs. One practical method for implementing multiplies is shown in FIG. **10**A. The values of A and B are complemented in A and B complement logic units **40**A and **40**B, respectively, is negative. The outputs of complement logic units **40**A and **40**B feed the multiplier front end **42**, which then always operates on positive values. The multiplier front end feeds a partial sum adder unit **44**, which is followed by complement end result logic **46** which sets the sign of the result correctly depending on the signs of A and B.

FIG. **10**B depicts a technique for processing negative-valued operands in accordance with the teachings herein. In FIG. **10**B it can be seen that ALUs **50**A and **50**B replace the complement logic units **40**A and **40**B, and are used to perform the complement operation during multiplication, as well as to check the end result and change the sign if necessary. The ALUs **50**A and **50**B can be used for other purposes at other times, such as for addition, subtraction, logical operations (for example, AND, OR, XOR, NOT) and for shifting. The ALUs **50**A and **50**B may also be used, as described above, to implement all or part of the adder block **44**.

While described in the context of the programmably configurable multiplier block **14**, it can be appreciated that other enhancements can be made as well to the DSP core **10**, such as the use of chainable registers to achieve wider register widths, splittable or combinable ALUs and the like. These enhancement techniques can also be used within the multiplier block **14** itself, in addition to providing the ALUs **14**B in place of the conventional dedicated adder circuitry of the prior art (see FIG. **4**). By example, FIG. **8** shows one suitable technique for achieving width scaling with two ALUs, wherein a multiplexer **20** operating under control of a Select input, or a simple gate such as an AND gate, is used for chaining together the two ALUs by selectively coupling the carry out terminal of ALU **1** to the carry in terminal of ALU **2**.

The use of this invention provides a number of advantages in the wireless terminal application, as different algorithms have different signal processing needs. For example, the Viterbi algorithm does not necessarily require 16-bit precision, as 8-bit precision may suffice. The use of this invention enables changes in precision in a programmable manner, and further enables the parallelism of the DSP core **10** to be changed in a dynamic manner. As an example, multiplication operations are required when performing metric calculations for the Viterbi algorithm, however for detecting the correct path through the trellis add/compare/select operations are required. The use of the teachings of this invention enables the DSP core to be programmably configured to perform the necessary multiplications, and to then be dynamically reconfigured on-the-fly to increase the number and parallelism of ALUs in order to perform the add/compare/select operations.

It is assumed that the plurality of additional ALUs **14**B made possible by the use of this invention are exploited by suitable programming tools and instructions, such as by the use of C++ callable intrinsics and/or intelligent compiler design. The teachings of this invention enable improvements in the operation of certain existing instruction types, as well as an ability to define new instruction types. These teachings are furthermore applicable to a wide variety of DSP architecture types, including but not limited to superscalar architectures and very long instruction word (VLIW) architectures. The teachings of this invention can be seen to provide a "coarse-grained" programmability alternative to the relatively fixed DSP architectures and the much finer-grained programmability inherent in the use of FPGAs. The teachings of this invention also avoid the problems inherent in ASIC-based designs, as circuit changes are made rapidly under program instruction control, and can provide a more cost-effective solution than many FPGA designs. The use of this invention also enables various enhancements and optimizations to be made to the multiplier front end, so as to accommodate and work with the use of the ALU(s) at the inputs and outputs of the multiplier front end.

Thus, while the invention has been particularly shown and described with respect to preferred embodiments thereof, it will be understood by those skilled in the art that changes in form and details may be made therein without departing from the scope and spirit of the invention.

What is claimed is:

1. A data processor, comprising a multiplier block having a multiplier front end for generating partial products from input operands, and a plurality of arithmetic logic units (ALUs) having inputs switchably coupled, in a first mode of operation, to first data sources comprised of outputs of said multiplier front end for adding together partial products received therefrom to arrive at a multiplication result, said inputs of said plurality of ALUs being switchably coupled, in a second mode of operation, to second data sources for performing at least one of arithmetic and logical operations on data received from said second data sources. add language

2. A data processor as in claim **1**, wherein said partial products have a width of n-bits, and where a width of said ALUs is one of n-bits or less than n-bits.

3. A data processor as in claim **1**, wherein said partial products have a width of 8-bits, and where a width of said ALUs is one of 8-bits or 4-bits.

4. A data processor as in claim **1**, wherein said partial products have a width of 16-bits, and where a width of said ALUs is one of 16-bits, 8-bits or 4-bits.

5. A data processor as in claim **1**, wherein said partial products have a width of 32-bits, and where a width of said ALUs is one of 32-bits, 16-bits, 8-bits or 4-bits.

6. A data processor as in claim **1**, wherein said partial products have a width of n-bits, where a width of said ALUs is less than n-bits, and where at least some of said plurality of ALUs are switchably coupled together to provide an n-bit wide ALU.

7. A data processor as in claim **1**, wherein said inputs of said ALUs are switchably coupled under control of a program instruction.

8. A method of operating a data processor, comprising:

providing a multiplier block having a multiplier front end for generating partial products from input operations; and

providing said multiplier block with a plurality of arithmetic logic units (ALUs); wherein

in a first mode of operation, said plurality of ALUs have inputs switchably coupled to first data sources com-

prised of outputs of said multiplier front end for adding together partial products received therefrom to arrive at a multiplication result; and

in a second mode of operation, said inputs of said plurality of ALUs are switchably coupled to second data sources for performing at least one of arithmetic and logical operations on data received from said second data sources.

add language

9. A method as in claim 8, wherein said partial products have a width of n-bits, and where a width of said ALUs is one of n-bits or less than n-bits.

10. A method as in claim 8, wherein said partial products have a width of 8-bits, and where a width of said ALUs is one of 8-bits or 4-bits.

11. A method as in claim 8, wherein said partial products have a width of 16-bits, and where a width of said ALUs is one of 16-bits, 8-bits or 4-bits.

12. A method as in claim 8, wherein said partial products have a width of 32-bits, and where a width of said ALUs is one of 32-bits, 16-bits, 8-bits or 4-bits.

13. A method is in claim 8, wherein said partial products have a width of n-bits, where a width of said ALUs is less than n-bits, and further comprising switchably coupling together at least some of said plurality of ALUs to provide an n-bit wide ALU.

14. A method as in claim 8, wherein said inputs of said ALUs are switchably coupled under control of a program instruction.

15. A method as in claim 8, wherein said plurality of ALUs comprise the same or additional ALUs coupled to inputs of said multiplier front end for changing a sign of said input operands.

16. A method as in claim 8, wherein switchably coupling employs reconfigurable signal routing logic.

17. A digital signal processor (DSP), comprising a DSP core having a register file, at least one arithmetic logical unit (ALU), and at least one multiplier block comprised of a multiplier front end for generating partial products from input operands, said multiplier block further comprising circuitry for adding together said partial products, said circuitry comprising a plurality of ALUs having inputs that are programmably coupled, in a first mode of operation, to first data sources comprised of outputs of said multiplier front end for adding together partial products received therefrom to arrive at a multiplication result, said inputs of said plurality of ALUs being programmably coupled, in a second mode of operation, to second data sources for selectively operating together in parallel for performing at least one of arithmetic and logical operations on data received from said second data sources, wherein said partial products have a width of n-bits, and where a width of individual ones of said plurality of ALU is one of n-bits or less than n-bits. — add language

18. A DSP as in claim 17, wherein said plurality of ALUs comprise the same or additional ALUs that are coupled to inputs of said multiplier front end for changing a sign of said input operands.

19. A DSP as in claim 17, and further comprising reconfigurable signal routing logic for providing data paths to and from said plurality of ALUs.

* * * * *